Московский физико-технический институт (ГУ)
Факультет инноваций и высоких технологий
Сложность вычислений, осень 2014
Возможные темы индивидуальных проектов

При выполнении любого проекта требуется написать оригинальный самодостаточный текст на русском языке на заданную тему. Принимаемые форматы уточняйте у своего семинариста, наиболее предпочитаемым и заведомо подходящим является файл PDF, набранный в TEX' е. В тексте должны быть даны все необходимые определения (кроме общеизвестных), а также сформулирована и доказана хотя бы одна нетривиальная теорема. Текст должен быть снабжён списком использованной литературы, содержащим хотя бы два источника. Более подробные инструкции даны для каждого вида заданий отдельно. Сроки выполнения работы: выбор темы проекта — **21 октября 2014 г.**, сдача предварительной версии — **18 ноября 2014 г.**, сдача окончательной работы — **16 декабря 2014 г.** За проект ставится от 0 до 12 баллов, эта оценка входит в итоговую оценку с весом 0.2, при этом для получения итоговой оценки отлично(8–10) необходимо сдать проект хотя бы на 5. По согласованию с семинаристом возможен выбор темы, отсутствующей в этом перечне. В качестве проекта допускается сделать перевод реферата или обзора с английского (или другого) языка, но в таком случае источник должен быть указан явно, а проект оценивается не больше, чем на 6 баллов. Прямые текстуальные заимствования не допускаются.

Обзорные проекты по теории. Нужно сделать обзор определений и результатов и хотя бы один из нетривиальных результатов доказать.

1. $\mathbf{EXP}$-полные задачи.
2. Задачи из $\mathbf{NP} \cap \mathbf{coNP}$, про которые неизвестно полиномиальных алгоритмов.
3. Задача об изоморфизме графов: полиномиальные алгоритмы для отдельных классов, лучшие алгоритмы для общего случая, класс $\mathbf{Gi}$ и полные задачи в нём.
4. Класс $\oplus\mathbf{P}$.
5. «Стивов класс» $\mathbf{SC}$ языков, распознаваемых за полиномиальное время на полилогарифмической памяти. Теорема: $\mathbf{RL} \subset \mathbf{SC}$.
6. Колмогоровская сложность с ограничением на вычислительные ресурсы.
7. Коммуникационная сложность: классы $\mathbf{P}^{\mathrm{cc}}$, $\mathbf{NP}^{\mathrm{cc}}$, $\mathbf{BPP}^{\mathrm{cc}}$ и т.д.
8. Классы $\mathbf{SAC}^n$ и $\mathbf{LOGCFL}$.

Изложение доказательства некоторой теоремы.

9. Теорема Блума: для некоторых задач не существует оптимальных программ, как бы ни мерялась сложность.
10. Теорема Вэлианта–Вазирани: если существует полиномиальный алгоритм для определения выполнимости формул, имеющих не более одного выполняющего набора, то $\mathbf{NP} = \mathbf{RP}$.

11. Теорема Импальяццо–Вигдерсона: если в классе $\mathbf{E}$ есть язык, не распознаваемый схемами субэкспоненциального размера, то $\mathbf{P} = \mathbf{BPP}$.

12. Теорема Разборова–Рудича о естественных доказательствах.

13. Теорема Хартманиса–Иммермана–Сьюэльсона: $\mathbf{E} \neq \mathbf{NE}$ тогда и только тогда, когда $\mathbf{NP} \setminus \mathbf{P}$ содержит разреженные языки.

14. $\mathbf{P} \neq \mathbf{polyL}$.

15. $\oplus \notin \mathbf{AC}^0$.

16. $\mod_p \notin \mathbf{ACC}^0(q)$ для различных простых p и q.

Обзоры сложности задач в некоторой области математики. Дана некоторая область математики. Нужно составить выборку задач из этой области и классифицировать их насколько возможно по различным сложностным классам: $\mathbf{P}$, $\mathbf{NP}$, $\mathbf{coNP}$, $\mathbf{PH}$, $\mathbf{PSPACE}$ и т.д. Хотя бы для одной из задач нужно доказать полноту в соответствующем классе. Хорошим стартовым пунктом может служить классификация задач в [1].

17. Биматричные игры.

18. Теория расписаний.

19. Алгебра и теория чисел.

20. Автоматы и формальные языки.

21. Теория гиперграфов.

22. Теория решёток.

23. Запросы к базам данным.

24. Пропозициональные модальные логики.

Исследовательские проекты. Требуется провести мини-исследование и изложить его результаты.

25. Истинность утверждения $\mathbf{P}^A = \mathbf{NP}^A$ для различных оракулов A.

26. Нахождение ошибки в одном из «доказательств» утверждения о (не)равенстве $\mathbf{P}$ и $\mathbf{NP}$, приведённых на странице `http://www.win.tue.nl/~gwoegi/P-versus-NP.htm`.

27. Исследование сложности какой-либо компьютерной или настольной игры. (Игра не должна содержать случайности, либо алгоритм должен заранее знать все розыгрыши случайных величин. Желательно, чтобы игрок был один, т.е. чтобы в игре требовалось выполнить некоторое задание. Пример такой игры — «Лягушки-непоседы» (“Hoppers”). Если в игре зафиксирован размер поля, то её нужно обобщить на произвольный размер).

Алгоритмические проекты. Требуется описать некоторый алгоритм, доказать его корректность и имплементировать его (возможно, в частном случае) на любом языке программирования. Текст отчёта должен включать в себя описание тестовых запусков.

28. **Покрытие множества**

    Дано множество M, а также набор его подмножеств $\{S_1, S_2, \dots, S_n\}$. Требуется найти минимальный набор из заданных подмножеств, такой, что их объединение равно (покрывает) M.

    Например, если $M = \{1, 2, 3, 4, 5\}$ и $S_1 = \{1, 2, 3\}$, $S_2 = \{2, 3, 4\}$, $S_3 = \{1, 3, 5\}$, то минимальный покрывающий набор состоит из множеств S_2 и S_3.

    Задание

    (a) Докажите, что проверка существования покрытия из не более чем k множеств является **NP**-полной.

    (b) Постройте алгоритм, дающий $\log n$-приближение, и имплементируйте его.

    (c) Предположим, что каждый элемент множества M присутствует не более чем в k подмножествах. Постройте алгоритм, дающий k-приближение (подсказка: сведите к задаче линейного программирования).

29. **Дерево Штейнера**

    Дан ненаправленный связный граф $G = (V, E)$, в нем выделено множество вершин V_0. Также имеются веса на ребрах $w : E \to \mathbb{R}_+$. Требуется найти дерево T минимального веса, покрывающее все вершины V_0.

    Задание

    (a) Докажите, что проверка существования дерева веса не более k является **NP**-полной.

    (b) Сведите задачу к метрической версии (в метрической версии исходный граф является полным и верно $w(x, z) \leqslant w(x, y) + w(y, z)$.

    (c) Постройте алгоритм, дающий 2-приближение для метрической версии задачи, а также имплементируйте его.

30. **Задача коммивояжера**

    Дан граф $G = (V, E)$ и веса на ребрах $w : V \to \mathbb{R}_+$. Требуется найти гамильтонов цикл минимального веса.

    Также выделяют отдельный случай задачи коммивояжера, когда граф полный и функция весов метрическая (то есть для любых трех вершин x, y и z выполнено $w(x, z) \leqslant w(x, y) + w(y, z)$).

Задание

(a) Докажите, что для стандартной задачи коммивояжера не существует константных алгоритмов приближения, если $\mathbf{P} \neq \mathbf{NP}$.

(b) Постройте алгоритм, дающий 2-приближение для метрической задачи коммивояжера на основе остовного дерева.

(c) Постройте алгоритм, дающий 1.5-приближение для метрической задачи коммивояжера на основе остовного дерева и паросочетания.

(d) Имплементируйте один из двух вышепостроенных алгоритмов. При выборе более простого алгоритма требуется написать более продвинутый интерфейс.

31. **Задача о рюкзаке**

Имеется набор из n предметов. У каждого предмета есть положительный вес w и стоимость c. Также дано неотрицательное число W — вместимость рюкзака. Требуется найти такое множество предметов M, чтобы оно помещалось в рюкзак, и суммарная стоимость предметов была максимальна. То есть:

$$\sum_{x \in M} w(x) \leqslant W$$
$$\sum_{x \in M} c(x) \to \max$$

Задание

Постройте полиномиальную схему приближения для данной задачи. То есть необходимо придумать и имплементировать алгоритм, который получает на вход экземпляр задачи о рюкзаке, а также произвольное (рациональное) $\varepsilon > 0$, и находит $(1+\varepsilon)$-приближенное решение. Алгоритм должен работать за полиномиальное время относительно размера исходной задачи и $1/\varepsilon$.

32. **Задача об оптимальном расписании**

Имеется множество работ J и множество машин M. Также задана функция $p : J \times M \to \mathbb{R}_+$. Значение p_{ij} означает время выполнение i-ой работы на j-ой машине.

Требуется построить распределение работ по машинам так, чтобы все работы были выполнены и чтобы конечное время выполнения всех работ было минимально. То требуется найти функцию $x : J \times M \to \{0, 1\}$ такую, что:

$$\sum_{j \in M} x_{ij} = 1, \quad \text{для всех } i$$
$$\max_{j \in M} \sum_i x_{ij} p_{ij} \to \min$$

Рассмотрим частный случай задачи, когда производительности всех машин одинаковы, то есть $p_{ij} = p_i$.

Задание

(a) Докажите, что задача об оптимальном расписании является **NP**-полной (подразумевается задача поиска расписания длительности не более k).

(b) Постройте алгоритм, дающий $4/3$-приближение для случая машин одинаковой производительности, а также имплементируйте его.

33. **Задача о наименьшей надстроке**

Пусть задано множество строк $A = \{\alpha_1, \alpha_2, \ldots, \alpha_n\}$ над конечным алфавитом. Требуется найти такую строку ω, что все строки из множества A являются подстроками ω и длина ω минимальна.

Задание

(a) Докажите, что задача является **NP**-полной (подразумевается задача поиска надстроки длины не больше k).

(b) Постройте алгоритм, дающий 4-приближение.

(c) Известно (но не доказано), что жадный алгоритм (который на каждом шаге находит строки с наибольшим перекрытием и склеивает их) дает 2-приближение. Имплементируйте его и убедитесь, что на коротких строках алгоритм действительно дает 2-приближение.

34. **Вершинное покрытие**

Известно, что для задачи вершинного покрытия существует простой алгоритм, дающий 2-приближение. Он состоит в том, чтобы найти максимальное паросочетание в графе и взять по вершине из каждого ребра.

Рассмотрим взвешенный вариант задачи. То есть дан граф $G = (V, E)$, а также функция $w : V \to \mathbb{R}_+$. Требуется найти множество минимального веса, покрывающее все ребра графа. То есть такое $U \subset V$, что для любого $(u, v) \in E$ верно $u \in U$ или $v \in U$ и сумма $\sum_{u \in U} w(u)$ минимальна.

Задание

Требуется построить алгоритм дающий 2-приближение для взвешенной задачи вершинного покрытия, а также имплементировать его.

Подсказка: рассмотрите частный случай графа, в котором $w(u) = c \cdot deg(u)$. Ответ для таких графов строится просто, остается декомпозировать исходную задачу на задачи такого вида.

35. **Выполнимость 3-КНФ.** Постройте и имплементируйте алгоритм, про который можно доказать следующее:

(a) Он распознаёт выполнимость 3-КНФ;

(b) В случае $\mathbf{P} = \mathbf{NP}$ он делает это за полиномиальное время.

Список литературы

[1] Гэри М., Джонсон Д. Вычислительные машины и труднорешаемые задачи, М.: Мир, 1982

[2] Vazirani, V. Approximation Algorithms, Springer, 2001